

To Modify an existing boxfill use:

```
box=a.getboxfill('AMIP_psl')

box.list() # Will list all the boxfill
box.projection='linear'
lon30={-180:'180W',-150:'150W',0:'Eq'}
box.xticlabels1=lon30
box.xticlabels2=lon30
box.xticlabels(lon30, lon30) # Will set them both
box.xmtics1=''
box.xmtics2=''
box.xmtics(lon30, lon30) # Will set them both
box.yticlabels1=lat10
box.yticlabels2=lat10
box.yticlabels(lat10, lat10) # Will set them both
box.ymtics1=''
box.ymtics2=''
box.ymtics(lat10, lat10) # Will set them both
box.dataawc_y1=-90.0
box.dataawc_y2=90.0
box.dataawc_x1=-180.0
box.dataawc_x2=180.0
box.dataawc(-90, 90, -180, 180) # Will set them all
box.xaxisconvert='linear'
box.yaxisconvert='linear'
box.xyscale('linear', 'area_wt') # Will set them both
box.level_1=1e20
box.level_2=1e20
box.color_1=16
box.color_2=239
box.colors(16, 239 ) # Will set them both
box.boxfill_type='linear' # 'linear' - compute or spec
# 'log10' - plot using log10
# 'custom' - use custom value

box.legend=None # Hold the legend values
box.ext_1='n' # Show left overflow arrow
box.ext_2='y' # Show right overflow arrow
box.exts('n', 'y' ) # Will set them both
box.missing=241 # Color index value range 0
```

There are two possibilities for setting the boxfill levels:

A) Levels are all contiguous (Examples):

```
box.levels=( [0,20,25,30,35,40], )
box.levels=( [0,20,25,30,35,40,45,50] )
box.levels=[0,20,25,30,35,40]
box.levels=(0.0,20.0,25.0,30.0,35.0,40.0,50.0)
```

B) Levels are not contiguous (Examples):

```
box.levels=( [0,20], [30,40], [50,60] )
box.levels=( [0,20,25,30,35,40], [30,40], [50,60] )
```

There are three possibilities for setting the fillarea color index
box.fillareacolors=([22,33,44,55,66,77])

```
box.fillareacolors=(16,19,33,44)
box.fillareacolors=None
```

There are three possibilities for setting the fillarea style (Ex)

```
box.fillareastyle = 'solid'
box.fillareastyle = 'hatch'
box.fillareastyle = 'pattern'
```

There are two ways to set the fillarea hatch or pattern indices

```
box.fillareaindices=(1,3,5,6,9,20)
box.fillareaindices=(7,1,4,9,6,15)
See using fillarea objects below!
```

Using the fillarea secondary object (Ex):

```
f=createfillarea('fill1')
To Create a new instance of fillarea use:
fill=a.createfillarea('new','quick') # Copies 'quick'
fill=a.createfillarea('new') # Copies 'default'
```

To Modify an existing boxfill use:

```
fill=a.getboxfill('def37')

box.fillareaindices=(7,fill,4,9,fill,15) # Set index
fill.list() # list fill
fill.style='hatch' # change st
fill.color=241 # change co
fill.index=3 # change st
```

Methods defined here:

```
__init__(self, parent, Gfb_name=None, Gfb_name_src='default', createGfb=0)
#####
#
# Initialize the boxfill attributes.
#
#####
```

colors(self, color1=16, color2=239)

dataw(self, dsp1=1e+20, dsp2=1e+20, dsp3=1e+20, dsp4=1e+20)

exts(self, ext1='n', ext2='y')

```
list(self)
#####
#
# List out boxfill graphics method members (attributes).
#
#####
```

rename = renameGfb(self, old_name, new_name)

```
#####
#
# Function:      renameGfb
#
# Description of Function:
#     Private function that renames the name of an existing
#     graphics method.
#
#
# Example of Use:
#     renameGfb(old_name, new_name)
#         where: old_name is the current name of boxfill
#         new_name is the new name for the boxfill
#
#####
```

script(self, script_filename=None, mode=None)

```
Function:      script                                     # Calls _vcs.s
```

```
Description of Function:
    Saves out a boxfill graphics method in Python or VCS so
    designated file.
```

Example of Use:

```
script(scriptfile_name, mode)
    where: scriptfile_name is the output name of the
           mode is either "w" for replace or "a" for
```

Note: If the the filename has a ".py" at the end
 Python script. If the filename has a ".scr"
 produce a VCS script. If neither extension
 default a Python script will be produced.

```
a=vcs.init()
box=a.createboxfill('temp')
box.script('filename.py')           # Append to a Python fil
box.script('filename.scr')         # Append to a VCS file "
box.script('filename','w')
```

xmtics(self, xmt1=",", xmt2="")

xticlabels(self, xtl1=",", xtl2="")

xyscale(self, xat=",", yat="")

ymtics(self, ymt1=",", ymt2="")

yticlabels(self, ytl1=",", ytl2="")

Properties defined here:

boxfill_type

```
get">get = _getboxfilltype(self)  
set">set = _setboxfilltype(self, value)
```

color_1

```
get">get = _getcolor_1(self)  
set">set = _setcolor_1(self, value)
```

color_2

```
get">get = _getcolor_2(self)  
set">set = _setcolor_2(self, value)
```

datawc_calendar

```
get">get = _getcalendar(self)  
set">set = _setcalendar(self, value)
```

datawc_timeunits

```
get">get = _gettimeunits(self)  
set">set = _settimeunits(self, value)
```

datawc_x1

```
get">get = _getdatawc_x1(self)  
set">set = _setdatawc_x1(self, value)
```

datawc_x2

```
get">get = _getdatawc_x2(self)  
set">set = _setdatawc_x2(self, value)
```

datawc_y1

```
get">get = _getdatawc_y1(self)  
set">set = _setdatawc_y1(self, value)
```

datawc_y2

```
get">get = _getdatawc_y2(self)  
set">set = _setdatawc_y2(self, value)
```

ext_1

```
get">get = _getext_1(self)  
set">set = _setext_1(self, value)
```

ext_2

```
get">get = _getext_2(self)  
set">set = _setext_2(self, value)
```

fillareacolors

```
get">get = _getfillareacolors(self)  
set">set = _setfillareacolors(self, value)
```

fillareaindices

```
get">get = _getfillareaindices(self)  
set">set = _setfillareaindices(self, value)
```

fillareastyle

```
get">get = _getfillareastyle(self)  
set">set = _setfillareastyle(self, value)
```

legend

```
get">get = _getlegend(self)  
set">set = _setlegend(self, value)
```

level_1

```
get">get = _getlevel_1(self)  
set">set = _setlevel_1(self, value)
```

level_2

```
get">get = _getlevel_2(self)  
set">set = _setlevel_2(self, value)
```

levels

```
get">get = _getlevels(self)  
set">set = _setlevels(self, value)
```

missing

```
get">get = _getmissing(self)  
set">set = _setmissing(self, value)
```

name

```
get">get = _getname(self)  
set">set = _setname(self, value)
```

projection

```
get">get = _getprojection(self)  
set">set = _setprojection(self, value)
```

xaxisconvert

```
get">get = _getxaxisconvert(self)  
set">set = _setxaxisconvert(self, value)
```

xmtics1

```
get">get = _getxmtics1(self)  
set">set = _setxmtics1(self, value)
```

xmtics2

```
get">get = _getxmtics2(self)  
set">set = _setxmtics2(self, value)
```

xticlabels1

```
get">get = _getxticlabels1(self)  
set">set = _setxticlabels1(self, value)
```

xticlabels2

```
get">get = _getxticlabels2(self)  
set">set = _setxticlabels2(self, value)
```

yaxisconvert

```
get">get = _getyaxisconvert(self)
set">set = _setyaxisconvert(self, value)
```

ymtics1

```
get">get = _getymtics1(self)
set">set = _setymtics1(self, value)
```

ymtics2

```
get">get = _getymtics2(self)
set">set = _setymtics2(self, value)
```

yticlabels1

```
get">get = _getyticlabels1(self)
set">set = _setyticlabels1(self, value)
```

yticlabels2

```
get">get = _getyticlabels2(self)
set">set = _setyticlabels2(self, value)
```

Data and other attributes defined here:

```
__slots__ = ['setmember', 'parent', 'name', 'g_name', 'xaxisconvert', 'yaxisconvert', 'levels', 'fillareac',
'ext_1', 'ext_2', 'missing', 'projection', 'xticlabels1', 'xticlabels2', 'yticlabels1', 'yticlabels2', 'xmtics1',
```

```
g_name = <member 'g_name' of 'Gfb' objects>
```

```
parent = <member 'parent' of 'Gfb' objects>
```

```
setmember = <member 'setmember' of 'Gfb' objects>
```

Functions

```
getGfbmember(self, member)
```

```
#####
#
# Function:      getGfbmember
#
# Description of Function:
#     Private function that retrieves the boxfill members from t
#     structure and passes it back to Python.
#
#
# Example of Use:
#     return_value =
#     getGfbmember(self, name)
#         where: self is the class (e.g., Gfb)
#                name is the name of the member that is being
#
#####
```

```

renameGfb(self, old_name, new_name)
#####
#
# Function:      renameGfb
#
# Description of Function:
#     Private function that renames the name of an existing boxfill graphics method.
#
#
# Example of Use:
#     renameGfb(old_name, new_name)
#         where: old_name is the current name of boxfill graphics method
#                new_name is the new name for the boxfill graphics method
#
#####

```

```

setGfbmember(self, member, value)
#####
#
# Function:      setGfbmember
#
# Description of Function:
#     Private function to update the VCS canvas plot. If the canvas is not
#     set to 0, then this function does nothing.
#
#
# Example of Use:
#     setGfbmember(self, name, value)
#         where: self is the class (e.g., Gfb)
#                name is the name of the member that is being updated
#                value is the new value of the member (or attribute)
#
#####

```

```

setmember = setGfbmember(self, member, value)
#####
#
# Function:      setGfbmember
#
# Description of Function:
#     Private function to update the VCS canvas plot. If the canvas is not
#     set to 0, then this function does nothing.
#
#
# Example of Use:
#     setGfbmember(self, name, value)
#         where: self is the class (e.g., Gfb)
#                name is the name of the member that is being updated
#                value is the new value of the member (or attribute)
#
#####

```

Data

StringTypes = (<type 'str'>, <type 'unicode'>)